AD-769 624

PAST, PRESENT AND FUTURE OF DEVELOPMENT, COMPUTATIONAL EFFICIENCY, AND PRACTICAL USE OF LARGE SCALE TRANSPORTATION AND TRANSHIPMENT COMPUTER CODES

A. Charnes, et al

Texas University

Prepared for:

Office of Naval Research

July 1973

AD-769 624

Unclassified
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Center for Cybernetic Studies University of Texas at Austin	2a. REPORT SECURITY CLASSIFICATION Unclassified
	2b. GROUP

3. REPORT TITLE

Past, Present and Future of Development, Computational Efficiency, and Practical Use of Large Scale Transportation and Transhipment Computer Codes

4. DESCRIPTIVE NOTES *(Type of report and, inclusive dates)*

5. AUTHOR(S) *(First name, middle initial, last name)*

A. Charnes
Fred Glover
David Karney
D. Klingman
Joel Stutz

6. REPORT DATE July, 1973	7a. TOTAL NO. OF PAGES ~~10~~ 22	7b. NO. OF REFS 35

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S) Center for Cybernetic Studies Research Report No. 131
b. PROJECT NO. N00014-67-A-0126-0008	
c. N00014-67-A-0126-0009	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Office of Naval Research (Code 434) Washington, D. C.

13. ABSTRACT

Three generations of computers have elapsed since the first satisfactory method for solving transportation and transhipment problems was devised. During this time many computational advances have taken place in developing computer codes to solve these problems. The primary purpose of this paper is to summarize these events and to do some crystal ball gazing to provide what we believe to be "best estimates" of future trends.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Networks						
Transportation problems						
Transhipment problems						
Linear programming						

Research Report
C.S. 131

PAST, PRESENT AND FUTURE OF DEVELOPMENT, COMPUTATIONAL EFFICIENCY, AND PRACTICAL USE OF LARGE SCALE TRANSPORTATION AND TRANSHIPMENT COMPUTER CODES

by

A. Charnes
Fred Glover *
David Karney
D. Klingman
Joel Stutz

July 1973

* Professor, University of Colorado

This research was partly supported by a grant from the Farah Foundation and by ONR Contracts N00014-67-A-0126-0008 and N00014-67-A-0126-0009 with the Center for Cybernetic Studies, The University of Texas. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES

A. Charnes, Director
Business-Economics Building, 512
The University of Texas
Austin, Texas 78712

Abstract

Three generations of computers have elapsed since the first satisfactory method for solving transportation and transhipment problems was devised. During this time many computational advances have taken place in developing computer codes to solve these problems. The primary purpose of this paper is to summarize these events and to do some crystal ball gazing to provide what we believe to be "best estimates" of future trends.

PAST

Approximately 200 years have elapsed since the French Academy of Sciences posed the civil engineering problem of "cutting and filling." Their formulation of the problem was not the same as the transportation problem as we know it today, but was the equivalent of a transportation problem in continuous form. The current formulation of a transportation problem was due to Kantorovich (1939), Hitchcock (1941), and Koopmans (1947).

Kantorovich showed in 1939 that a class of problems closely related to the transportation problem had a remarkable variety of applications. These were concerned typically with the allotment of tasks to machines whose costs and rates of production vary by task and machine type. Kantorovich gave a useful but incomplete algorithm for solving such problems. Later, in 1942, he studied both discrete and continuous versions of this problem and in 1948, along with Gavurin wrote an applicational study on the capacitated transportation problem.

Hitchcock developed an incomplete algorithm in 1941, which exploited special properties of the transportation problem to find starting solutions. Koopmans (1947) independently arrived at the same problem in connection with his work as a member of the Combines Shipping Adjustment Board. He and Reiter discussed the problem from an economic efficiency analysis viewpoint and pointed out the analogy between it and the classical Maxwell-Kirchhoff electrical network problem. Because of their work, the problem is often referred as the Hitchcock-Koopmans Transportation Problem.

The first generally satisfactory method for solving the general class of transportation models was due to G.B. Dantzig in 1949. This method is sometimes called the Row-Column Sum Method [4] or the MODI method [7]. Charnes and Cooper (1954) later wrote an explanation (dubbed the Stepping-Stone Method) of the simplex steps involved in the Row-Column Sum Method. With the advent of a method for solving the transportation problem came numerous methods for securing starting bases. Two of the methods commonly referenced are the Northwest-Corner Rule [7] and the Vogel Approximation Method [31] (often referred to as VAM). Of all the start methods developed, VAM became the one most used for hand calculations due to the excellent start it provides. Thus in the folklore VAM is considered the best procedure for both computer and hand calculations.

After the development of the Row-Column Sum Method, the transportation model, with integer parameters, rapidly became the chief "solvable" integer linear programming problem due to the integer extreme point property. Also a survey by L.W. Smith, Jr. in 1956 indicated that at least half of the linear programming applications used this model. Some reasons for the surprising concentration on problems of this kind, particularly in applications, are:

(1) Answers to "large" problems can be easily computed by <u>hand</u>, which is an impossible task for general linear programming problems of similar dimensions. Also, integer solutions were immediately attainable.

(2) It is possible to approximate many linear programs by transportation problems.

(3) A number of seemingly unrelated linear programs have been found to

be equivalent to transportation problems.

(4) Computer codes were developed as early as 1952 for solving transportation problems.

(5) Business executives can understand the transportation model, leading to increased demand for its applications in practical settings.

Subsequent to the development of the simplex based Row-Column Sum Method, Ford and Fulkerson (1955), developed a primal-dual method for solving transportation problems. Somewhat earlier Gleyzal [18] developed a method similar to the primal-dual method.

It is interesting to note that Dantzig and Ford and Fulkerson concluded on the basis of hand calculations that the primal-dual method was superior in efficiency to the Row-Column Sum Method. This conclusion was also supported by Flood (1961) on computer codes. Consequently, this conclusion became part of the folklore.

The first computer code for solving transportation problems was based on the Row-Column Sum Method and in terms of current jargon is called a primal transportation code. About 1952 such a code was developed by the George Washington University Logistics Research Project in conjunction with the Computation Laboratory of the National Bureau of Standards [33]. This code, which was designed for use on the Bureau of Standards Eastern Automatic Computer, was further improved by the NBS Computation Laboratory. The code was capable of solving problems with at most 600 nodes with a pivot time of 3 plus minutes per pivot. Current pivot times for problems of this size are 6-10 milliseconds on the CDC 6600, UNIVAC 1108, and IBM 360/65 computers.

The incore-out-of-core primal code by Dennis (1958) is one of the first codes to be described in detail in the literature. Dennis' paper is also one of the first to study different criteria for selecting pivot elements. Unfortunately, his study principally involved only one problem of size 30 origins by 260 destinations. The best solution time was 9.6 minutes on the Whirlwind computer.

Another primal code was developed in the late fifties by Glicksman, Johnson and Eselson (1959). This code was developed for the UNIVAC I for solving "thin rectangular" problems and was also an incore - out-of-core code. The code solved a 15 origin by 488 destination problem in 24 minutes. This is approximately 200 times slower than current incore codes.

Also, during the late fifties a number of transportation codes were developed using Kuhn's Hungarian Method, implemented primarily on IBM computers. These codes include the one due to Flood (1961) using his proof of the Konig-Egervary Theorem. Codes based on Ford and Fulkerson's primal-dual algorithm were also beginning to be implemented, such as the IB-TFL code (1958). The code of Flood and the IB-TFL code were compared on a problem with 29 origins and 116 destinations on an IBM 704. Their times were 3.03 and 3.17 minutes, respectively. Current solution time would probably be 2 seconds.

Following these developments, there was a histus of half a dozen years during which little was visibly accomplished in the development of improved solution methods or computer implementations. From an algorithmic standpoint, it was widely believed that no significant refinements remained to be discovered. In retrospect, this attitude seems surprising, particularly in view of the paucity of experimentation to determine the computational strengths and weaknesses of alternative approaches. Then, in the later sixties and more particularly in the early seventies, a new surge of interest in network methods and applications came about, leading to a number of surprises for those steeped in the notions of a decade earlier.

It is to these more recent developments that we now turn.

PRESENT

As already intimated, the early special purpose primal and primal-dual codes were capable of solving only small problems, were quite slow, and were not extensively tested. Beginning with the latter half of the sixties, several codes have been jointly developed by mathematical programmers and systems analysts who have performed extensive experimentation on various algorithmic rules. In particular, recent code development and/or computational studies have been performed by Barr, Glover, and Klingman (1972) on an out-of-kilter network code, Bennington (1971-72) on a non-extreme point network code, Boeing (1966) on an out-of-kilter code, Clasen (1966) on an out-of-kilter code, Control Data Corporation (1970) on an out-of-kilter code, Glover, Karney, Klingman, and Napier (1970-1972) on a primal transportation code, Glover, Karney, and Klingman (1970-1972) on a dual transportation code, Glover, Karney, and Klingman (1973) on a primal transhipment code, Klingman Napier, and Ross (1973) on parameter sensitivity analysis, Klingman, Napier, and Stutz (1973) on a network generator code, Lee (1968) on various solution procedures, Srinivasan and Thompson (1970-1972) on a primal transportation code, Texas Water Development Board (1968) on an out-of-kilter code, and UNIVAC (1973) on an out-of-kilter code. It should be noted that all of these codes are in-core codes coded in FORTRAN.

The primal transportation and the primal transhipment codes by Glover, Karney, and Klingman [13, 15] utilize the augmented predecessor index method [14],which elaborates on Johnson's "triple-label method"[20] by providing an efficient method for characterizing successive basis trees with minimal relabeling. The augmented predecessor list structure has been a major contributor to the improvements in the computational efficiency of solution algorithms. With its use, the primal transportation code by Glover, Karney, and Klingman [15] executes a pivot on a 600 node problem in 6 milliseconds compared with the early breakthrough (1952) of reducing the time per pivot to 3 minutes. While this reduction is largely due to improvements in computers and the in-core nature of these codes, this is not the whole reason. For instance, the first accelerated primal transportation code developed by Srinivasan and Thompson [34] employed a list structure for preceeding in a forward direction through a spanning tree similar to Dennis' procedure. Upon comparing solution times of the Srinivasan and Thompson code with the Glover, Karney, and Klingman code on the same problems and machine, the efficiencies of the augmented index structure became apparent. Srinivasan and Thompson recoded using the augmented list structure and cut their solution times by more than half.

The code development and comparison of Srinivasan and Thompson [34] provides an important computational analysis of several primal start procedures and pivot criteria. The purpose of this study was to determine a design for an in-core uncapacitated primal transportation and assignment code which optimally combines start procedures and pivot criteria for maximum solution efficiency. The study disclosed that the best start method is the "modified row minimum start" procedure and the best pivot selection criterion is the "row most negative rule." This pivot rule was also found to be best by Dennis [8]. Maximum problem size solved was 350 nodes (origins plus destinations). This node limitation is due to the fact that the code was all in-core and stored a complete cost matrix. The average solution time on 175 origin by 175 destination transportation and assignment problems was 7.8 seconds.

The code development and comparison by Glover, Karney, Klingman, and Napier (1970-1972) performed similar analyses on a broad profile of dense and nondense problems. The underlying algorithm was specially designed for solving both capacitated and uncapacitated problems with nondense cost matrices (i.e., transportation problems where some cells may not be allowable). This study also found the modified row minimum start and row most negative pivot rule to be best, thus casting doubt on the folklore of the superiority of VAM starts. In addition, the study compared 100 problems on this primal code to several other codes including Clasen's SHARE code (1966), the Glover, Karney, and Klingman dual code (1970), and the state-of-the-art linear programming code OPHELIE/LP. This comparison revealed that the specially designed primal code was at least eight times faster than the SHARE and dual codes, and 150 times faster than OPHELIE/LP. Thus the old folklore about the superiority of out-of-kilter methods, and a new folklore among computer service divisions about equivalence of general purpose and special purpose solution codes for transportation and transhipment problems were upended.

The largest problems solved by the Glover, Karney, Klingman and Napier study were 1000 origin by 1000 destination problems with an average solution time of 17 seconds. This study also tested the primal code on four computers, IBM 360/65, UNIVAC 1108, CDC 6400, and CDC 6600 in order to provide insights into conclusions based on comparing times on different machines and compilers. It was discovered that standard guidelines concerning the relative efficiencies of different computers were completely misleading, since the primal code ran only 10%-12% faster on the CDC 6600 than on the UNIVAC 1108 and the IBM 360/65 differing substantially from the estimates one would obtain by comparing instruction execution times of the machines.

Motivated by the fact that out-of-kilter codes were found to be substantially slower than the special primal code, Barr, Glover, and Klingman (1972) developed an improved version of the out-of-kilter method which was subsequently coded. This code was found to be only 40% slower (on the same problems and machine) than the primal transportation code of Glover, Karney, and Klingman on transportation problems. This code was also compared against Clasen's SHARE code, Boeing's code, and the Texas Water Development Board code and found to be at least six times faster than the best of these (which differed from problem to problem). The study also examined a total of 215 capacitated and uncapacitated transhipment problems demonstrating the superiority of the improved version of the out-of-kilter code over the other out-of-kilter codes in all cases. The largest problems solved were 1500 node transhipment problems. The mean solution time was 34 seconds.

Still more recently, Glover, Karney, and Klingman (1973) have developed a general primal transhipment code. Computational comparison of this code with the out-of-kilter code by Barr, Glover, and Klingman reveals that the primal code is 30% faster on transhipment problems. This is rather startling since the Barr, Glover, and Klingman code is probably the fastest out-of-kilter code in the world and conventional wisdom has it that labeling techniques are inherently more efficient than simplex techniques. The primal transhipment code was also tested against an out-of-kilter code by Bennington (1971-72) and found to be eight times faster. This computational study also shows the superiority of the new primal transhipment code in terms of central memory requirements for storing network data. Specifically, the out-of-kilter codes discussed earlier require 6-10 arc-length arrays and 4-7 node-length arrays as compared to 3 arc-length arrays and 8 node-length arrays for the primal code. The substantially increased problem size that can be accommodated by the new primal code is illustrated in the study by the solution of an 8000 node problem.

In addition to recent code development efforts and computational comparisons, two computational studies on the effects of parameter values have been conducted on transportation problems. The first study, by Srinivasan and Thompson (1970-71), examined the effect on solution time of rectangularity, the number of significant digits in uniformly distributed cost coefficients, and the number of significant digits in the supply and demand values. The study analyzed 600 transportation problems with a maximum size of 350 nodes. The main finding is that solution time is not particularly sensitive to the supply and demand ranges, but is quite dependent on cost ranges. The second study, by Klingman, Napier, and Ross (1973), performed a detailed examination of the effects of problem dimensionality on solution times. The study included over 1000 randomly generated problems with 185 different combinations of number of origins, number of destinations and number of variables (not all cells being considered admissible). Every problem was solved using three starting procedures. Over 10,000 pieces of data were analyzed, providing numerous insights into the computational effects of the number of constraints, the degree of "rectangularity," and the number of variables. An unexpected finding of the study was that no single starting procedure dominates another; rather, the efficiency of a starting procedure varies with problem structure.

These studies on the effect of parameter values and code comparisons repeatedly reinforced the conclusion that in order for researchers to compare their codes in a meaningful way it is necessary that they use exactly the same problems. This is due to the fact that, even if two randomly generated problems have the same parameter values, a generator inherently builds structure into the problems, particularly transhipment and transportation problems in which some arcs do not exist. This point is underscored in the computational study by Barr, Glover, and Klingman and has been further demonstrated by the comparison of the codes due to Bennington; Glover, Klingman, and Karney; and Srinivasan and Thompson which yielded unexpected results when tested on the same problems and the same computer.

To enable researchers to meaningfully compare their solution codes, Klingman, Napier and Stutz [24] developed a code which generates assignment problems, and capacitated and uncapacitated transportation and transhipment problems. In addition to producing structurally different classes of transhipment problems, the code permits the user to vary structural characteristics within a class. By means of this code, researchers can generate identical transhipment problems (independent of the computer). Advantages of the transhipment generator, which is available with documentation to researchers for a nominal handling charge, are its ease of use (requiring only two data cards per transhipment problem) and the standardization of its output (generating problems for use by other codes in SHARE input format). In addition, the latter part of the documentation provides the user with the data on 40 assignment, transportation, and transhipment problems varying in size from 200 nodes to 8,000 nodes and from 1,300 arcs to 35,000 arcs. The objective function value and solution time for these problems are also provided for Clasen's SHARE code, Boeing's code, Barr, Glover and Klingman out-of-kilter code, and the Glover, Karney and Klingman primal transhipment code.

FUTURE

In spite of the major recent gains in the development and testing of network codes, significant avenues remain to be explored. In particular, all the codes currently in vogue are in-core codes, all are coded in FORTRAN, most have not fully exploited problem size capability of third generation computers, and all of them can only solve pure network problems as opposed to generalized or weighted network problems.

In the near future, we will undoubtedly see large scale in-core-out-of core network codes developed which are capable of solving network problems of almost unlimited size. In this regard, there is an in-core-out-of-core primal transhipment code under development by Karney and Klingman which has not been fully streamlined but which has solved a problem for the Naval Personnel Laboratory in San Diego with 2400 nodes and 450,000 arcs on the IBM 360/65 and CDC 6600 in 66 and 65 minutes, respectively. Also, there is a primal generalized network code being developed by Glover, Klingman, and Stutz utilizing extensions of the augmented predecessor method [17] that has proved successful for the pure network codes. In addition, we shall probably see codes developed in other languages (e.g., ALGOL and APL) in order to rigorously determine which language is best in network applications. Assembly language codes are also quite conceivably in the offing.

While the transhipment generator [24] is a start towards helping benchmark these codes, we believe that a bureau needs to be established to enable standardized comparisons. We have been informed by Richard Jackson that the National Bureau of Standards in conjunction with various researchers, is considering the feasibility of establishing a service facility to accomplish this. The benefit of establishing such a service is apparent; however, numerous problems must be overcome. For example, Input/Output formats of codes and the timing of codes must be standardized. While such matters may appear to be quite simple, they are not. To illustrate, every code benchmarked by the authors, e.g., those due to Clasen, Boeing, the Texas Water Development Board, General Motors, Bennington, and Srinivasan and Thompson, used a different input format, and considerable effort was required to accommodate these differences. Also, while a valid criterion for the timing of in-core codes is quite easy to define (namely central processing time exclusive of data input and output), establishing an acceptable measure for the timing of in-core-out-of-core codes in a multi-processing environment is far more complex. In any case, we believe some criteria need to be developed.

Other short range future developments which we foresee include:

a) development of network computer systems similar to general linear programming systems. These systems will include such things as a command language (which allows the user to add, delete, and modify arcs), matrix generators, report generators, user subroutine control of the system or any component, and interactive coupling with data base management information systems. The Control Data Corporation NETFLOW [29] system and the UNIVAC UKILT-1100 [35] system are forerunners of such systems.

b) establishment of numerous special purpose integer programming codes using efficient network codes as the main computational vehicle, e.g., plant location codes, fixed charge network codes, integer generalized network codes, constrained network codes, multi-commodity (integer) network codes, constrained generalized network codes, and multi-commodity (integer) generalized network codes.

Looking farther into the future, we anticipate the following as possible developments:

a) an efficient graph computer language which allows a user to write special purpose network codes in half a day. A forerunner is the GROPE language at the University of Texas.

b) network and related optimization codes which modify themselves- for example, computer network codes which "learn" how to efficiently solve particular types of problems through experience in solving them. (Preliminary investigations of this type have now been going on for more than a decade.)

c) multi-page linear programming codes which use special purpose codes (e.g., network codes) to solve pages (components) which have a special structure.

BIBLIOGRAPHY

[1] Barr, R. S., F. Glover, and D. Klingman, "An Improved Version of the Out-of-Kilter Method and a Comparative Study of Computer Codes." To appear in Mathematical Programming.

[2] Bennington, G. E., "An Efficient Minimal Cost Flow Algorithm," O. R. Report 75, North Carolina State University, Raleigh, North Carolina, (June 1972).

[3] Charnes, A., and W. W. Cooper, "The Stepping Stone Method of Explaining Linear Programming Calculations in Transportation Problems," Management Science (October, 1954), 49-69.

[4] ________, ________, Management Models and Industrial Applications of Linear Programming, 2 vols., (New York: John Wiley and Sons, Inc., 1961).

[5] Clasen, R. J., "The Numerical Solution of Network Problems Using the Out-of-Kilter Algorithm," RAND Corporation Memorandum RM-5456-PR, Santa Monica, California, (March, 1968).

[6] Dantzig, G., Chapter XXIII, Activity Analysis of Production and Allocation, Edited by T. C. Koopmans, John Wiley and Sons, (1951).

[7] ________, Linear Programming and Extensions, (Princeton, New Jersey: Princeton University Press, 1963).

[8] Dennis, J. B., "A High-Speed Computer Technique for the Transportation Problem," Journal of Association for Computing Machinery, Vol. 8, (1958), 132-153.

[9] Flood, M. M., "A Transportation Algorithm and Code," Naval Research Logistics Quarterly, 8 (1961), 257-276.

[10] Ford, L. R. and D. Fulkerson, "A Primal-Dual Algorithm for the Capacitated Hitchcock Problem," Naval Research Logistics Quarterly, 4, 1 (1957) 47-54.

[11] Fulkerson, D. R., "An Out-of-Kilter Method for Solving Minimal Cost Flow Problems," J. Soc. Indust. Appl. Math, 9 (1961), 18-27.

[12] Glickman, S., L. Johnson and L. Eselson, "Coding the Transportation Problem," Naval Research Logistics Quarterly, 7 (1960), 169-183.

[13] Glover, F., D. Karney, and D. Klingman, "Implementation and Computational Study on Start Procedures and Basis Change Criteria for a Primal Network Code," Research Report CS 136, Center for Cybernetic Studies, University of Texas, Austin (1973).

[14] _______, _______, _______, "The Augmented Predecessor Index Method for Locating Stepping Stone Paths and Assigning Dual Prices in Distribution Problems," Transportation Science, 6, 1 (1972), 171-180.

[15] _______, _______, _______, and A. Napier, "A Computational Study on Start Procedures, Basis Change Criteria, and Solution Algorithms for Transportation Problems." To appear in Management Science.

[16] _______, _______, _______, "Double-Pricing Dual and Feasible Start Algorithms for the Capacitated Transportation (distribution) Problem," University of Texas at Austin (1970).

[17] _______, _______, and J. Stutz, "Extensions of the Augmented Predecessor Index Method to Generalized Network Problems," Transportation Science, 7, 4, (1973).

[18] Gleyzal, A. N., "An Algorithm for Solving the Transportation Problem," Journal of Research of the National Bureau of Standards, 54, 4 (1955), 213-216.

[19] Hitchcock, F. L., "The Distribution of a Product from Several Sources to Numerous Localities," Journal of Math. Phys. 20 (1941), 224-230.

[20] Johnson, Ellis, "Networks and Basic Solutions," Operations Research, 14, 4 (1966), 619-623.

[21] Kantorovich, L. V., "On the Translocation of Masses," Compt. Rend. Acad. Sci. U.S.S.R., 37 (1942), 199-201.

[22] ________, and M. K. Gavurin, "The Application of Mathematical Methods to Problems of Freight Flow Analysis," Akademia Nauk SSSR, (1949).

[23] Klingman, D., A. Napier, and G. Ross, "A Computational Study on the Effects of Problem Dimensions on Solution Time for Transportation Problems," Research Report CS 135 Center for Cybernetic Studies, University of Texas, Austin (1973).

[24] ________, ________, and J. Stutz, "NETGEN - A Program for Generating Large Scale (Un) Capacitated Assignment, Transportation, and Minimum Cost Flow Network Problems." To appear in Management Science.

[25] Koopmans, T. C., Activity Analysis of Production and Allocation, Cowles Commission Monograph No. 13 (New York: John Wiley and Sons, Inc., 1951).

[26] ________, and S. Reiter, "A Model of Transportation," Activity Analysis of Production and Allocation, Cowles Commission Monograph 13, Wiley, (1951), 222-259.

[27] Kuhn, H. W., "The Hungarian Method for the Assignment Problem," Naval Research Logistics Quarterly, 2, (1955), 83-97.

[28] Lee, S., "An Experimental Study of the Transportation Algorithms," Master's Thesis, Graduate School of Business, University of California at Los Angeles (1968).

[29] "Network Flow Routine," YIM/FOCUS Library Number: H3-CODA NETFLOW; Control Data Corporation, Software Mfg. and Distribution, 215 Moffett Park Drive, Sunnyvale, California.

[30] "Out-of-Kilter Network Routine," SHARE Distribution 3536, SHARE Distribution Agency, Hawthorne, New York (1967).

[31] Reinfeld, N. V. and W. R. Vogel, Mathematical Programming (Englewood Cliffs, N. J.: Prentice-Hall, Inc., 1958).

[32] Smith, L. W., Jr., "Current Status of the Industrial Use of Linear Programming," Management Science 2 (1956), 156-158.

[33] Stanley, E. D., and L. Gainen, "Linear Programming in Bid Evaluation," Naval Research Logistics Quarterly, I (1954), 48-54.

[34] Srinivasan, V. and G. L. Thompson, "Benefit-Cost Analysis of Coding Techniques for the Primal Transportation Algorithm," ACM, 20 (1973), 194-213.

[35] "UKILT-1100 Programmer Reference Manual," UNIVAC, Data Processing Division, Roseville, Minnesota.